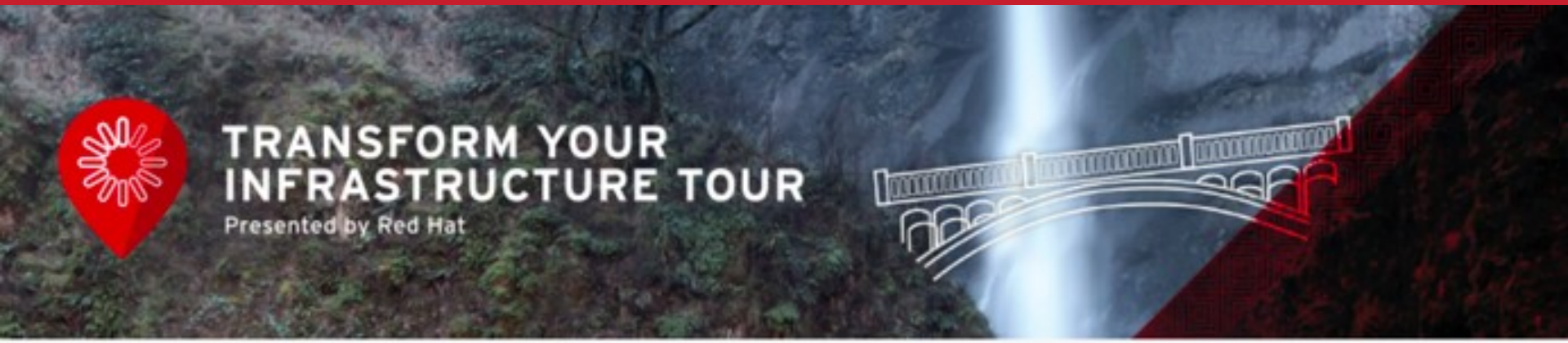
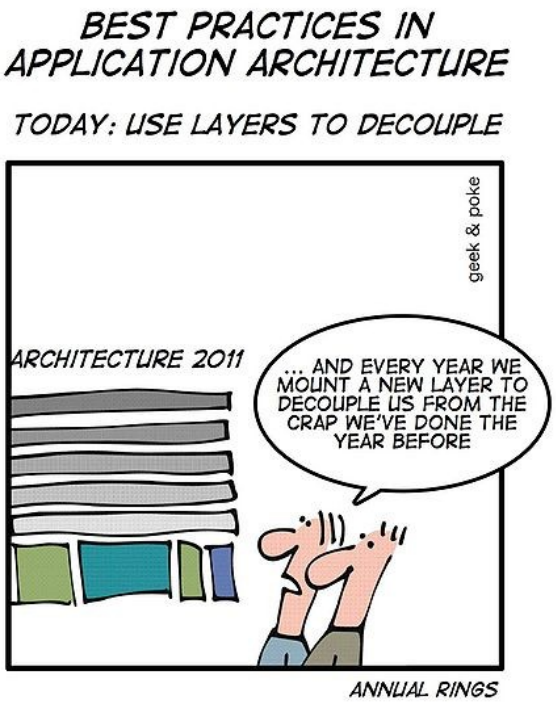
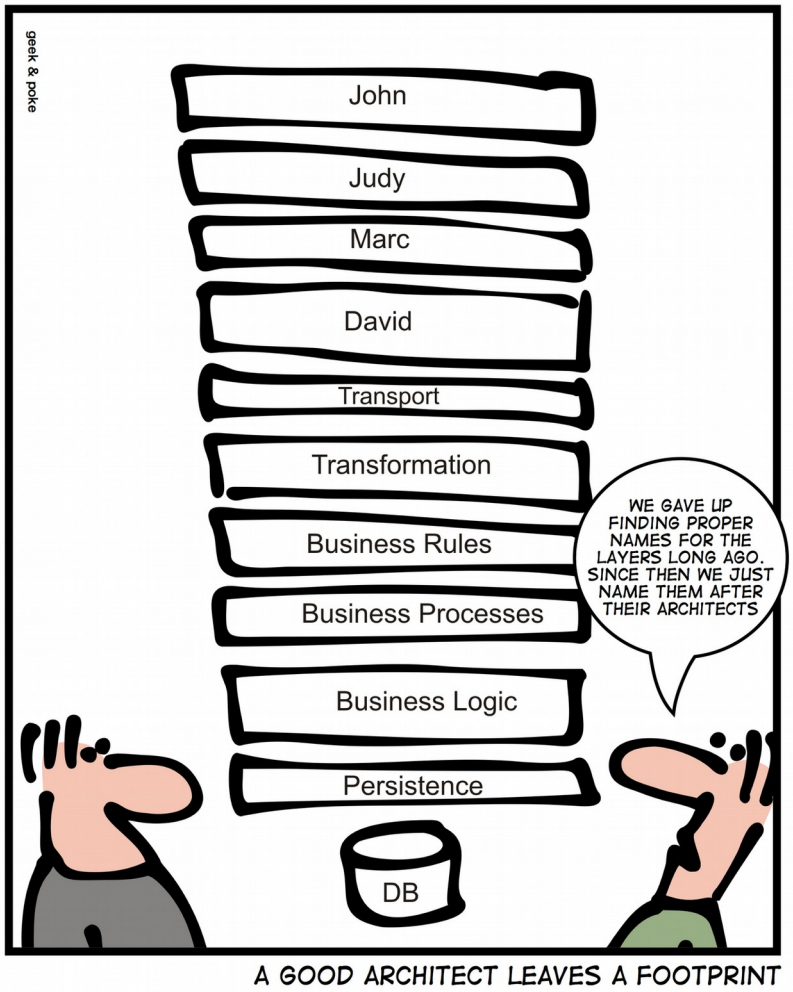
Deliver Microservices with RHEL Atomic & Docker

Ravi Srinivasan
Senior Consultant, Red Hat India




BEST PRACTICES IN APPLICATION ARCHITECTURE
TODAY: USE LAYERS TO DECOUPLE
geek & poke
ARCHITECTURE 2011
... AND EVERY YEAR WE MOUNT A NEW LAYER TO DECOUPLE US FROM THE CRAP WE'VE DONE THE YEAR BEFORE
ANNUAL RINGS


geek & poke
John
Judy
Marc
David
Transport
Transformation
Business Rules
Business Processes
Business Logic
Persistence
DB
WE GAVE UP FINDING PROPER NAMES FOR THE LAYERS LONG AGO. SINCE THEN WE JUST NAME THEM AFTER THEIR ARCHITECTS
A GOOD ARCHITECT LEAVES A FOOTPRINT

MONOLITH VS MICROSERVICES

“Don’t even consider microservices unless you have a system that's too complex to manage as a monolith”

- Martin Fowler

Source: http://martinfowler.com/bliki/MicroservicePremium.html

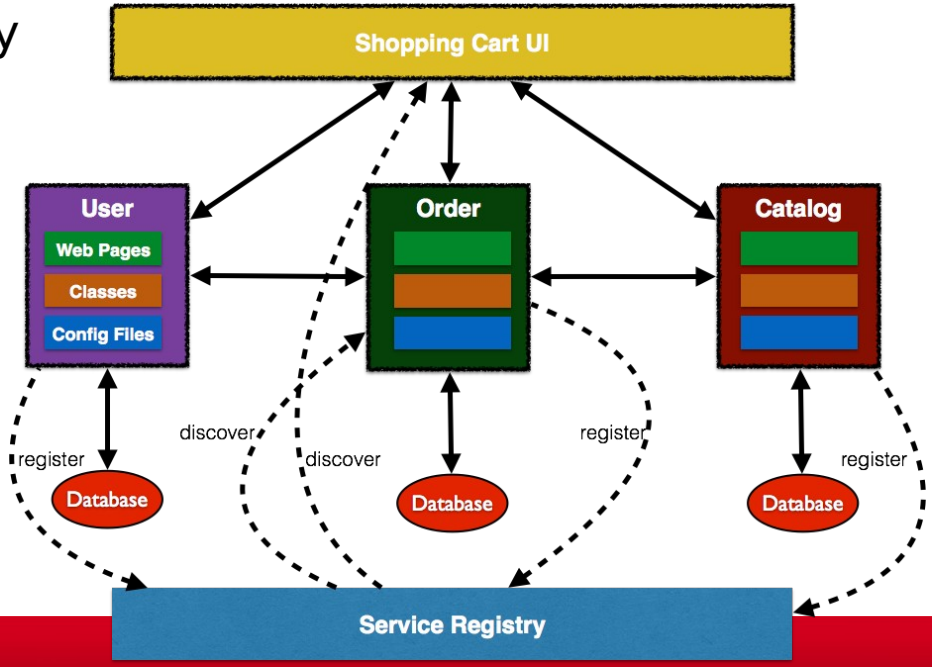
MICROSERVICES

"...an approach to developing a single application as a suite of small services, each running in its own process and communicating with lightweight mechanisms, often an HTTP resource API."

- Martin Fowler

MICROSERVICES ADVANTAGES

- Fast to develop, easier to maintain and understand
- Starts faster, speeds up deployments
- Local change can be easily deployed, great enabler of CD
- Fault isolation
- Services can be scaled independently
- Local changes can be deployed easily
- Eliminates any long-term commitment to a technology stack
- Flexible choice of technology

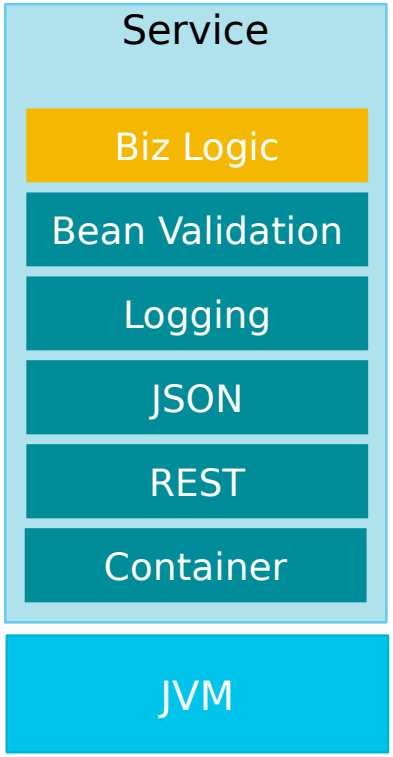


USE CASES

- STATELESS, SCALE OUT type workloads
  - API Gateways
  - Caching Servers/Proxy Servers/Load Balancers
  - Stateless Web Applications
- Parallel processing – Map/Reduce Type workloads
- Demand Based Scaling of workloads
- DevOps CI/CD type workflows/Automated Testing
- Case Study – E-Gov Services platform
  - Seasonal demand challenges
  - Massive user base (Scalability)
  - Complex integration scenarios – many Ad-Hoc
  - Must be Cloud ready
  - Security

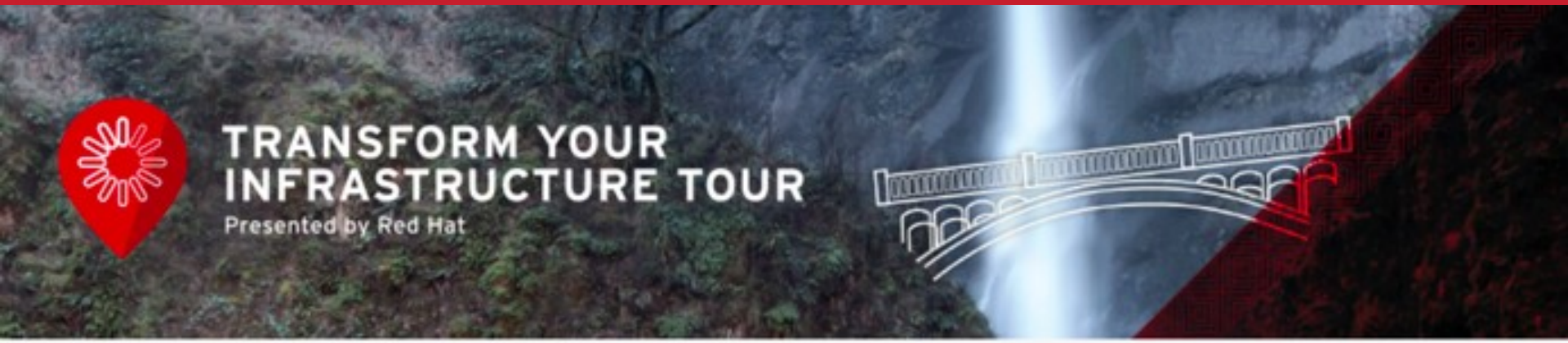
ERA OF “CONTAINERLESS” APPLICATION

- Truly containerless
  - NodeJS
  - Vert.x
- Container as a dependency (embedded)
- Self-contained executable fat jar
- (Opinionated) frameworks to help
  - Dropwizard (Jetty)
  - Spring Boot (Tomcat, Jetty or Undertow)
  - WildFly Swarm (Wildfly)



redhat.

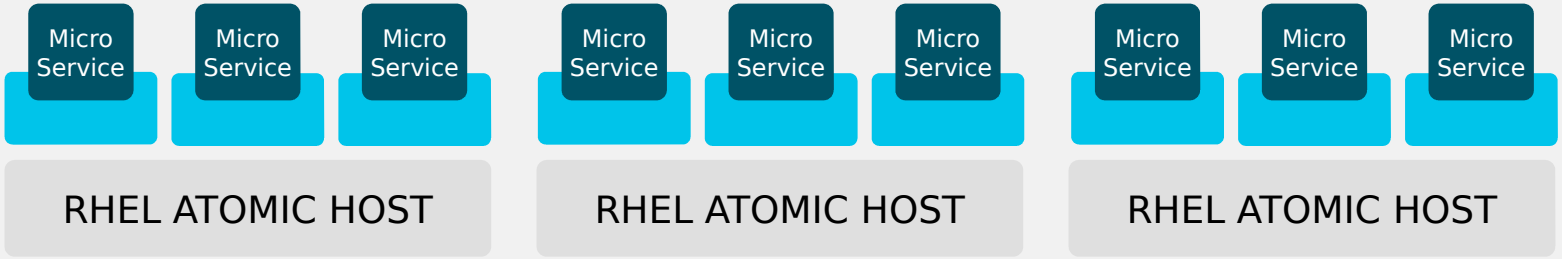
Red Hat's Microservices Vision



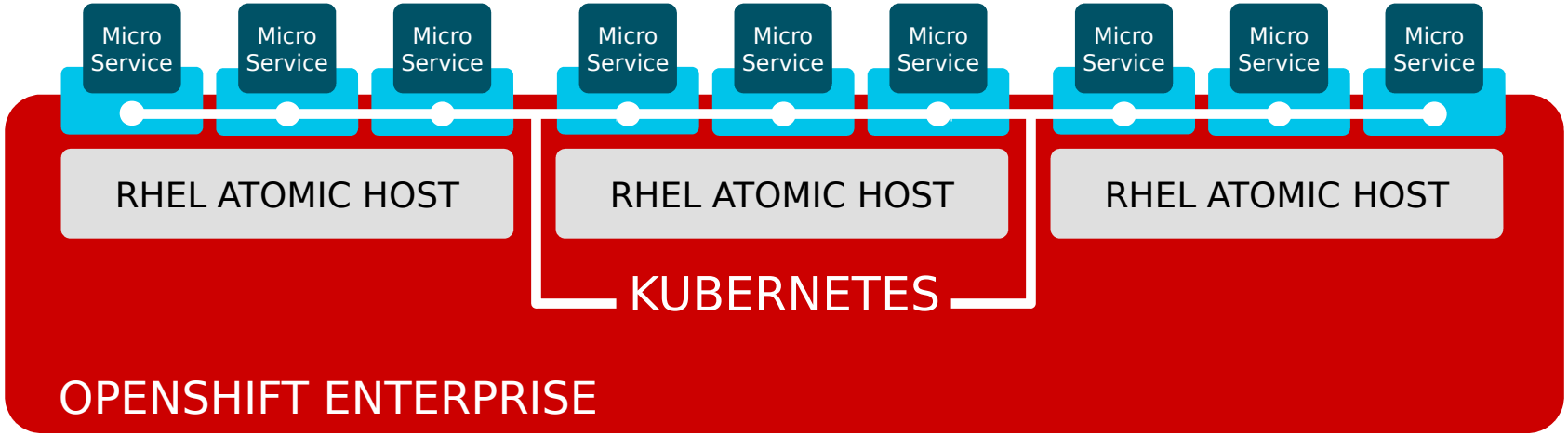
redhat | intel

MICROSERVICE + OS CONTAINER

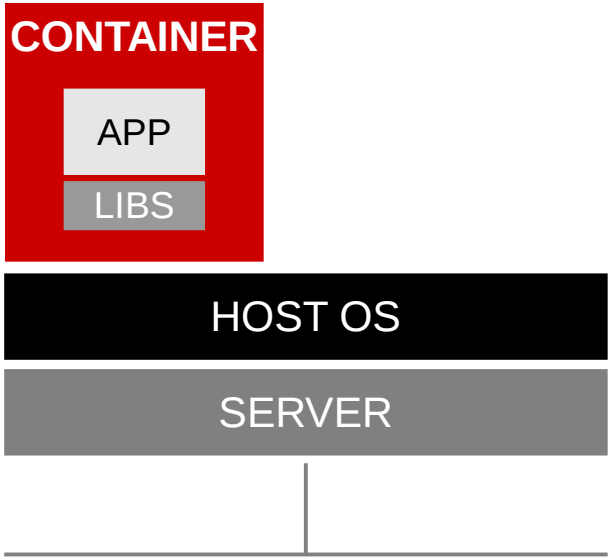
RHEL ATOMIC HOST:
THE WAY TO RUN DOCKER



KUBERNETES FOR ORCHESTRATION

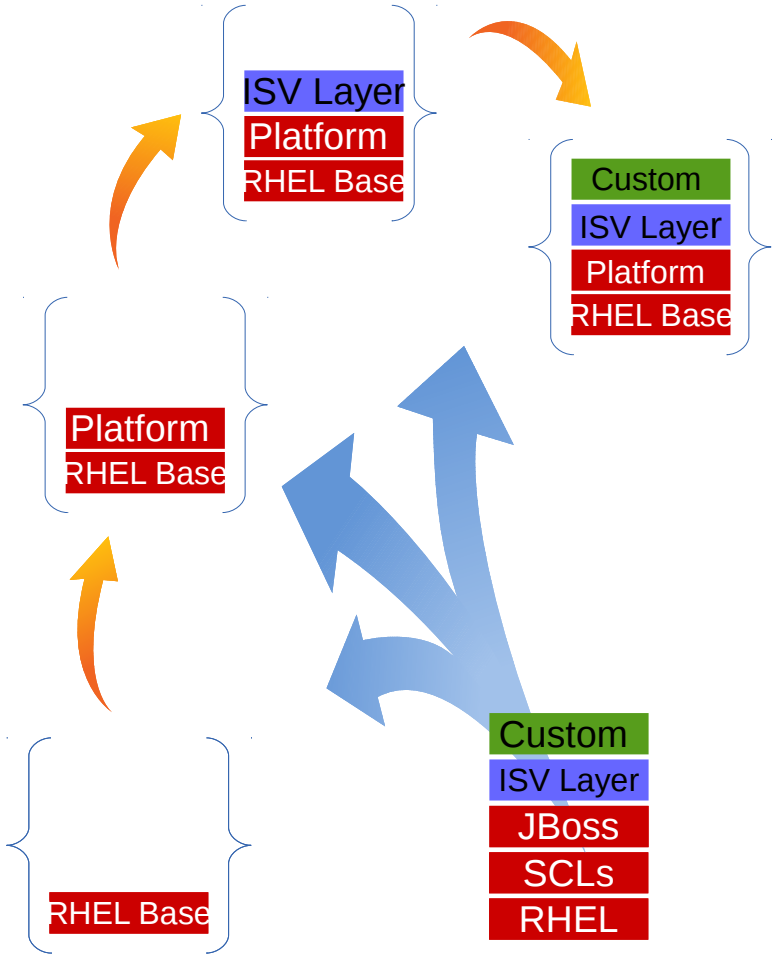


- Containers are:
  - software packaging concept that typically includes an application and all of its runtime dependencies
  - easy to deploy
  - portable across host systems
  - means to isolate application from host OS
- Virtualize the App, not the OS



Container Layering

- New images can be created by adding layers
- Layering model allows for specialization
- Base image and select number of platform layers provided by Red Hat
- ISV images form the base of the RHEL ecosystem
- Stack optimized for individual application with minimal packaging per layer



RHEL Atomic

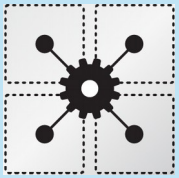
ISOLATION WITH LINUX CONTAINERS

- Isolating applications on host operating system
- Security
- Portability across host systems

CONTAINER FORMAT WITH DOCKER

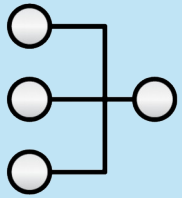
- Interface for communications, configuration, data persistence, provisioning
- Content agnostic
- Infrastructure agnostic

ORCHESTRATION WITH KUBERNETES



- Orchestrate at scale
- Define app topologies
- Handle networking
- Manage state
- Schedule across hosts

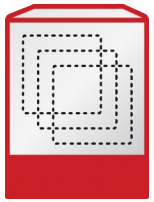
RPM OSTREE FOR ATOMIC TREES



- New deployment model for RHEL, uses RHEL RPMs under the covers
- Single shot updates and rollbacks
- Applications run inside containers

RHEL Atomic Host

IT IS RED HAT ENTERPRISE LINUX



Inherits the complete hardware ecosystem, military-grade security, stability and reliability for which Red Hat Enterprise Linux is known.

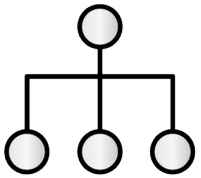
OPTIMIZED FOR CONTAINERS

MINIMIZED FOOTPRINT

Minimized host environment tuned for running Linux containers while maintaining compatibility with Red Hat Enterprise Linux.

SIMPLIFIED MAINTENANCE

Atomic updating and rollback means it's easy to deploy, update, and rollback using imaged-based technology.

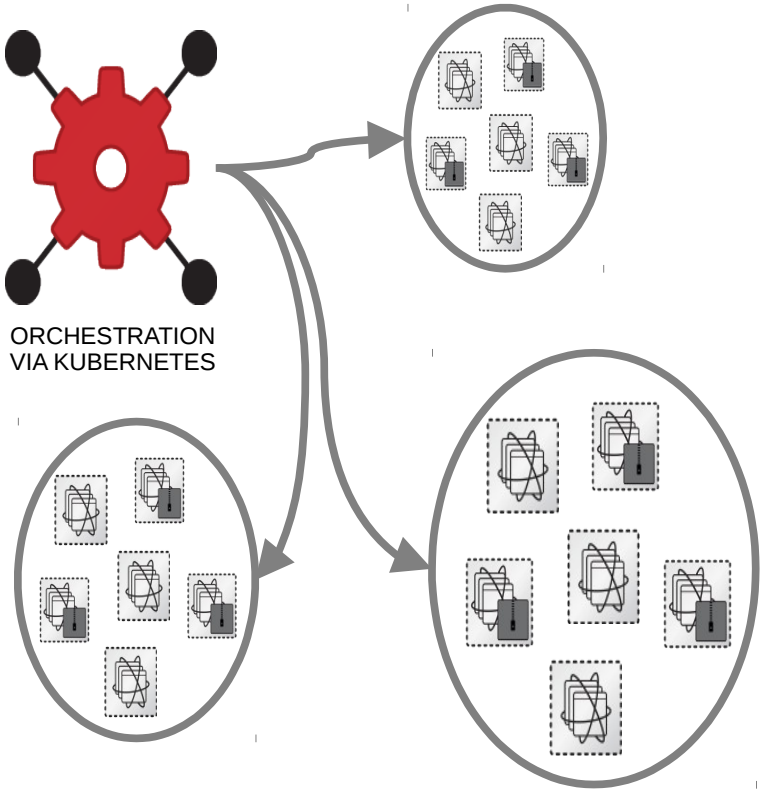


ORCHESTRATION AT SCALE

Build composite applications by orchestrating multiple containers as microservices on a single host instance.

CONTAINER ORCHESTRATION, SCHEDULING, AND MANAGEMENT VIA KUBERNETES

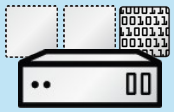
Critical for building containerized application infrastructure



- Orchestrate application services that span multiple containers across multiple Linux hosts
- Schedule containers across multiple hosts in desired topology
  - Enable manual and automated scaling up & down
- Manage container lifecycle with declarative model for health management to detect and restart on failure

Atomic Host Deployment

PHYSICAL SYSTEM ISO

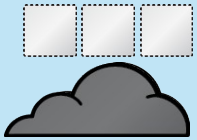


- Anaconda installation
- Kickstart installation
- PXEboot installation

CERTIFIED HYPERVISORS

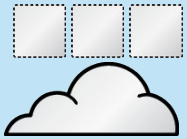
- RHEL (Qcow2 for KVM based HV)
- RHEV (OVA import for KVM based HV)
- VMware (OVA for ESX based HV)
- Microsoft (VHD for Hyper-V HV)

PRIVATE CLOUDS



- RHEL Open StackPlatform (Qcow2 for KVM based HV)

PUBLIC CLOUDS



- RHEL Atomic Host images available on select public clouds via cloud access
- Amazon Web Services (AWS)
- Google Compute Engine (GCE)

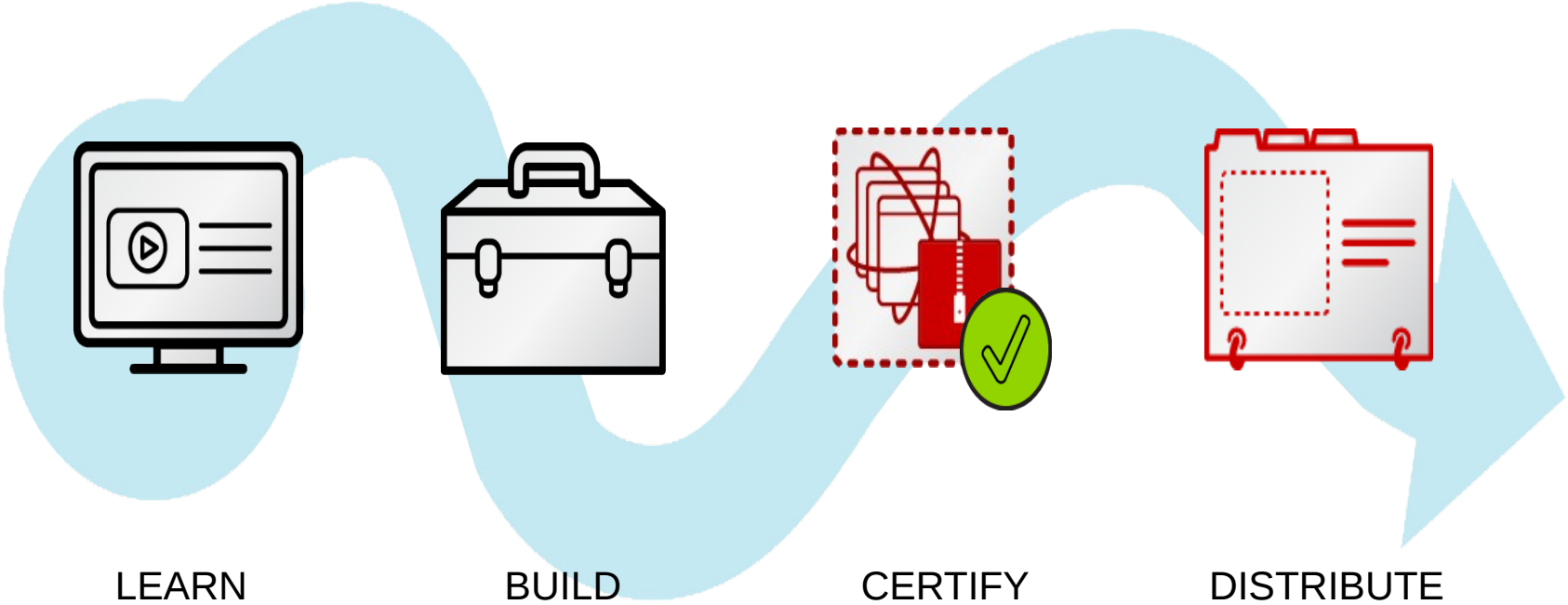
SIMPLIFYING CONTAINER ADOPTION

RED HAT CONNECT for technology partners

RED HAT CONTAINER DEVELOPMENT KIT (CDK)

RED HAT CONTAINER CERTIFICATION

RED HAT CONTAINER REGISTRY



LEARN

BUILD

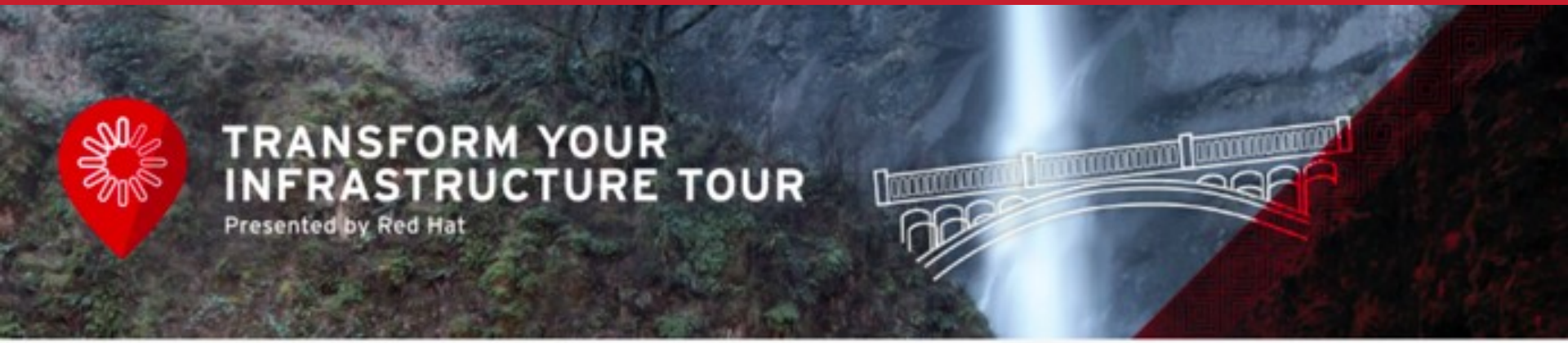
CERTIFY

DISTRIBUTE

https://connect.redhat.com

redhat.

Demo 1: Distributed CDR Processing
Demo 2: Polyglot Microservices applications



redhat | intel

Thank You

rsriniva@redhat.com
github.com/rsriniva/itwb-demo